PATENT ABSTRACTS OF JAPAN

(11)Publication number : 07-093269
(43)Date of publication of application : 07.04.1995

(51)Int.Cl. G06F 15/16
G06F 9/45

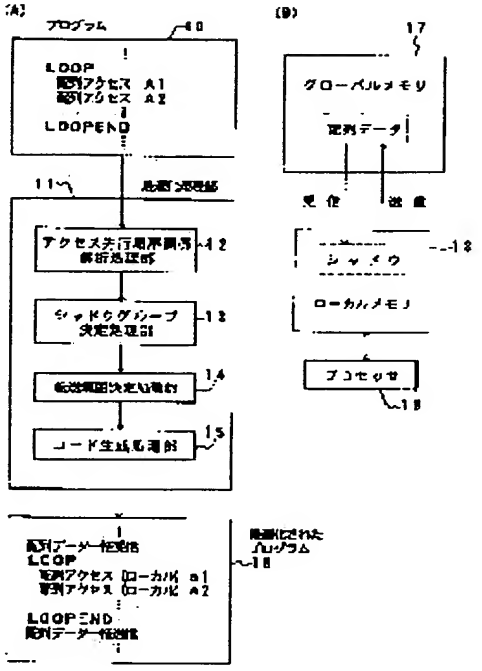
(21)Application number : 05-292919
(22)Date of filing : 24.11.1993

(71)Applicant : FUJITSU LTD
(72)Inventor : IWASHITA HIDETOSHI

(30)Priority
Priority number : 05185065 Priority date : 27.07.1993 Priority country : JP

(54) BATCH PROCESSING METHOD FOR DATA TRANSFER

(57)Abstract:
PURPOSE: To reduce overhead and to reduce a data transfer amount by classifying arrangement access inside a loop into groups based on preceding order relation and generating batch access corresponding to each group outside the loop.
CONSTITUTION: When there is access A1, A2... of arrangement data at a global memory 17 concerning a program 10 of an optimizing target, this access is classified into groups for making shadow common, and a transfer range and the shape of the shadow are decided. Then, an instruction for simultaneously receiving the required arrangement data from the global memory 17 all together for each of shadow is added before the loop, and an instruction for simultaneously transmitting the arrangement data as the processing result to the global memory 17 all together is added after the loop. Inside the loop, the shadow at a local memory 18 is accessed. Thus, it is not necessary for each loop to transfer the data of an element unit with the global memory 17, the number of times of data transfer is reduced due to batch transfer, and the transfer amount can be reduced.



EGAL STATUS

Date of request for examination]	20.12.1999
Date of sending the examiner's decision of rejection]	
Kind of final disposal of application other than the examiner's ecision of rejection or application converted registration]	
Date of final disposal for application]	
Patent number]	3261239
Date of registration]	14.12.2001
Number of appeal against examiner's decision of rejection]	
Date of requesting appeal against examiner's decision of rejection]	
Date of extinction of right]	

(19)日本国特許庁（ＪＰ）　(12) 公開特許公報（A）　(11)特許出願公開番号

特開平7－93269

(43)公開日　平成７年(1995)４月７日

(51)Int.Cl.[6]	識別記号	庁内整理番号	ＦＩ	技術表示箇所
Ｇ０６Ｆ　15/16	４３０　Ａ	7429－5L		
9/45				
		9292－5B	Ｇ０６Ｆ　9/44　３２２　Ｇ	

審査請求　未請求　請求項の数２　ＯＬ　（全 13 頁）

(21)出願番号　特願平5－292919

(22)出願日　平成５年(1993)11月24日

(31)優先権主張番号　特願平5－185065

(32)優先日　平５(1993)７月27日

(33)優先権主張国　日本（ＪＰ）

(71)出願人　000005223
富士通株式会社
神奈川県川崎市中原区上小田中1015番地

(72)発明者　岩下　英俊
神奈川県川崎市中原区上小田中1015番地
富士通株式会社内

(74)代理人　弁理士　小笠原　吉義　（外２名）

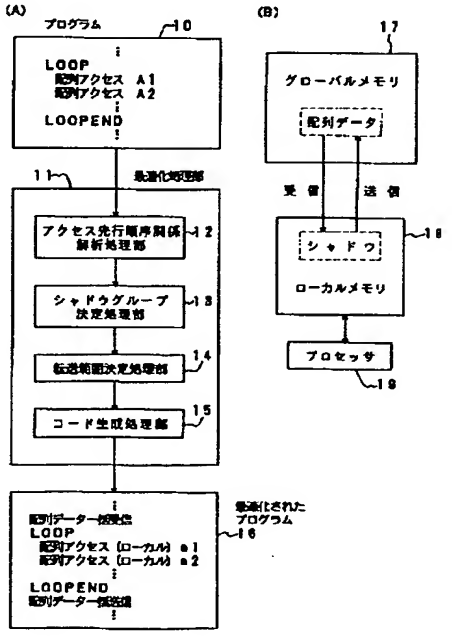
(54)【発明の名称】　データ転送の一括化処理方法

(57)【要約】

【目的】メモリが遠近２階層にモデル化できる計算機システムにおけるデータ転送の一括化処理方法に関し，ループ内のデータ転送を一括化してループ外に出すことにより，転送回数および転送総量を削減することを目的とする。

【構成】グローバルメモリ17にある配列データのアクセスを含むループ手続きを持つプログラム10について，そのアクセスの先行順序関係を解析し，シャドウを共通化するグループに分類する。そして，各グループに属するアクセスについて，それぞれアクセス要素の集合和を求め，データを転送する範囲を決定する。その範囲の配列データを一括アクセスする命令コードをループ外に生成し，最適化されたプログラム16を出力する。

本発明の原理説明図



1

【特許請求の範囲】

【請求項１】　１または複数のプロセッサ(19)と，そのプロセッサ(19)が高速にアクセスできる第１のメモリ(18)と，大容量または他のプロセッサと共用するデータを保持する第２のメモリ(17)とを備え，前記第２のメモリ(17)が保持するデータを前記第１のメモリ(18)に転送して処理する計算機システム上で動作するプログラム(10)を最適化する処理方法であって，前記第２のメモリ(17)にある配列データのアクセスを含むループ手続きが与えられたとき，そのアクセスの先行順序関係を解析する処理過程(12)と，アクセスの先行順序関係を解析した結果に基づいて，ループ内の配列アクセスを，前記第１のメモリ(18)におけるデータの一時的な格納場所の共通化が必要なグループに分類する処理過程(13)と，各グループに属するアクセスについて，それぞれアクセス要素の集合和を求め，データを転送する範囲を決定する処理過程(14)と，前記各グループに属する転送範囲の配列データを一括アクセスする命令コードをループ外に生成する処理過程(15)とを有することを特徴とするデータ転送の一括化処理方法。

【請求項２】　１または複数のプロセッサ(19)と，そのプロセッサ(19)が高速にアクセスできる第１のメモリ(18)と，大容量または他のプロセッサと共用するデータを保持する第２のメモリ(17)とを備え，前記第２のメモリ(17)が保持するデータを前記第１のメモリ(18)に転送して処理する計算機システム上で動作するプログラム(10)を最適化する処理方法であって，前記第２のメモリ(17)にある配列データのアクセスを含むループ手続きが与えられたとき，そのアクセスの先行順序関係を解析する処理過程(12)と，アクセスの先行順序関係を解析した結果に基づいて，ループ内の配列アクセスを，前記第１のメモリ(18)におけるデータの一時的な格納場所の共通化が必要なグループに分類する処理過程(13)と，各グループに属するアクセスについて，ループ内で参照するアクセス要素の集合うち，参照に先立って定義されるアクセス要素の集合を受信転送範囲から除外し，各グループごとに受信転送範囲および送信転送範囲についてのアクセス要素の集合和を求め，データを転送する範囲を決定する処理過程(14)と，前記受信転送範囲の配列データを一括アクセスする命令コードをループの前に生成し，前記送信転送範囲の配列データを一括アクセスする命令コードをループの後に生成する処理過程(15)とを有することを特徴とするデータ転送の一括化処理方法。

【発明の詳細な説明】

【０００１】

【産業上の利用分野】本発明は，階層型のメモリを持つ並列計算機等において，遠方の配列データのアクセスを含むループ手続きが与えられたとき，アクセスを一括化してループ外に出すことにより，プログラムの最適化を実現したデータ転送の一括化処理方法に関する。

2

【０００２】並列計算機，ネットワーク接続された分散計算機環境，大容量外部メモリ装置を使用する計算機等では，一般に遠近２階層にモデル化されるメモリを使用する。このような階層型のメモリシステムにおいて，オーバヘッドの削減のためには，メモリ間のデータ転送回数および全体的なデータ転送量を減らすことが必要となる。

【０００３】

【従来の技術】図１３は本発明に関係する２階層メモリのモデル説明図，図１４は本発明の課題説明図である。

【０００４】並列計算機，ネットワーク接続された分散計算機環境，大容量外部メモリ装置を使用する計算機等では，図１３に示すようにメモリが遠近２階層にモデル化できる。図中の８０はプロセッサ，８１はプロセッサ８０が高速に直接アクセスできるローカル(Local) メモリ空間，８２はローカルメモリ空間８１を介してアクセスされるデータを保持するグローバル(Global)メモリ空間である。グローバルメモリ空間８２は，仮想的なものであってもよく，また，例えば分散メモリ型並列計算機では，物理的には他のプロセッサのローカルメモリであってもよい。

【０００５】図１３に示すような計算機構成では，グローバルメモリ空間８２とローカルメモリ空間８１との間のデータ転送は，１要素ずつ行うよりも一括化して転送回数を減らすほうが，立ち上がりのオーバヘッドが削減され効率がよい。また，可能な限り転送量を減らすほうがよい。

【０００６】しかし，従来，例えば図１４（ａ）に示すようなプログラムのループにおいて，グローバルメモリ空間にある配列データＡ（ｉ）等を参照または定義する命令があるような場合に，グローバルメモリ空間８２とローカルメモリ空間８１との間で必要最小限のデータだけを一括して転送することは考えられていなかった。そのため，データ転送回数が多くなる，または転送総量が多くなるという問題があった。なお，以下の説明において，グローバルメモリ空間８２にある配列データは英大文字で始まる配列名〔例：Ａ（ｉ）〕で表し，ローカルメモリ空間８１上の配列データは英小文字で始まる配列名〔例：ｃ（ｉ）〕で表す。

【０００７】

【発明が解決しようとする課題】ところで，ループにおけるデータをできるだけ大きな単位で扱う技術として，いわゆるベクトル計算機における最適化処理方法がある。ここで用いられるベクトル化技術では，アクセスするデータの先行順序関係を解析して，できるだけ大量なデータのベクトル計算を効率よく実行できるような命令コードの生成を行っている。

【０００８】前述のように，グローバルメモリ空間８２とローカルメモリ空間８１との間のデータ転送は，１要素ずつ行うよりも一括化して転送回数を減らすほうが，

3

立ち上がりのオーバヘッドが削減され効率がよい。そこで，ベクトル化技術を流用することを考える。ベクトル化技術を流用してデータの一括転送を図ることにより，２階層メモリにモデル化されるシステムにおいても，ある程度のデータ転送の一括化は可能であると考えられる。

【０００９】例えば，図１４（ａ）に示すようなプログラムをベクトル化すると，図１４（ｂ）に示すようになる。図１４（ｂ）において，ｃ（１：１００）は，ローカルメモリ空間８１における配列ｃの１から１００までの要素を表す。Ａ（１：１００）は，グローバルメモリ空間８２における配列Ａの１から１００までの要素を表す。他の配列表記も同様である。

【００１０】まず，図１４（ａ）に示すプログラムを，図１４（ｂ）に示すように既存の方法でベクトル化し，これをもとに，図１４（ｃ）に示すようにデータの受信／送信命令を付加する。すなわち，グローバル配列Ａの参照を，グローバルメモリ空間８２からローカルメモリ空間８１の一時域(tmp1)への転送(receive)と，一時域(tmp1)の参照に置き換え，また，グローバル配列Ａの定義を，ローカルメモリ空間８１における一時域(tmp2)の定義と，一時域(tmp2)からグローバルメモリ空間８２への転送(send)に置き換える。

【００１１】しかし，この方法では，データ依存関係によってベクトル化できない場合には，データ転送の一括化はできない。例えば図１４（ｄ）に示すようなプログラムの場合である。このプログラムでは，配列Ａ（１）の参照と配列Ａ（２）の定義が同じループ内で行われ，次のループで配列Ａ（２）が参照される。したがって，配列Ａ（２）の定義が，配列Ａ（２）の参照より先に実行されるが，ベクトル化するとこの順序が逆転することになるので，ベクトル化はできない。すなわち，この例の場合，一時域の適当な共通化によって，転送の一括化は可能であると考えられるが，ベクトル化技術の延長では実現は難しい。

【００１２】また，ベクトル化では，主に単一のループをその対象とするので，データ転送の効率化に十分な要素量の一括化ができない。一般にベクトル化では数十要素で十分であるが，データ転送では数千～数万要素を一括化してはじめて十分な効率が得られることが多い。特に，ループが多重の場合，多重ループ全体について一括化する技術が必要とされる。

【００１３】本発明は，一時変数をグループ分けし，グループごとに共通化することにより，ベクトル化できないループについても転送の一括化を可能にすることを目的とする。また，多重ループについてもデータ転送の一括化を図り，プログラムの高速実行を可能とすることを目的とする。さらにまた，転送データ量の削減を図ることを目的とする。

【００１４】

4

【課題を解決するための手段】図１は本発明の原理説明図である。図１（Ａ）に示すプログラム１０は，図１（Ｂ）に示すような計算機システムで実行されるプログラムであって，グローバルメモリ１７が保持する配列データのアクセスを含むループ（ＬＯＯＰ）手続きを持つプログラムである。プロセッサ１９は，グローバルメモリ１７の配列データを通常の演算命令でダイレクトに読み込んで処理したり，グローバルメモリ１７へダイレクトに書き出したりすることはできず，一旦ローカルメモリ１８に転送して，ローカルメモリ１８内で処理し，書き出す際には，処理結果をローカルメモリ１８からグローバルメモリ１７へ転送する。ローカルメモリ１８における，グローバルメモリ１７の写しをシャドウという。

【００１５】最適化処理部１１は，プログラム１０を最適化し，実行の高速化を図る手段である。本発明では，各処理部により以下の処理を行う。アクセス先行順序関係解析処理部１２は，グローバルメモリ１７にある配列データのアクセスを含むループ手続きを持つプログラム１０が与えられたとき，そのアクセスの先行順序関係を解析する。

【００１６】シャドウグループ決定処理部１３は，アクセスの先行順序関係を解析した結果に基づいて，ループ内の配列アクセスを，シャドウを共通化するグループ（これを，シャドウグループという）に分類する。すなわち，先行順序が特定の関係にある複数の配列アクセスは必ず同一のグループに属するように分類する。

【００１７】転送範囲決定処理部１４は，各シャドウグループに属するアクセスについて，それぞれアクセス要素の集合和を求め，データを転送する範囲を決定する。特に，請求項２記載の発明では，転送範囲決定処理部１４は，ループ内で参照するアクセス要素の集合のうち，参照に先立って定義されるアクセス要素の集合を受信転送範囲から除外して，受信転送範囲および送信転送範囲についてのアクセス要素の集合和を求め，データを転送する範囲を決定することにより，さらに受信転送データ量の削減を図る。

【００１８】コード生成処理部１５は，各シャドウグループに属する転送範囲の配列データを一括アクセスする命令コードをループ外に生成し，ループ内にあるグローバルメモリ１７への配列アクセスは，シャドウのアクセスに変形する。これより，最適化されたプログラム１６が生成され，出力される。

【００１９】

【作用】本発明は，図１（Ｂ）のようなシステムにおいて，グローバルメモリ１７に対するアクセスの一括化によるオーバヘッドの削減と，データ転送量の削減を図ったものであり，そのために，ループ内の配列アクセスを先行順序関係を基にグループ分けし，各グループに対応する一括アクセスを，ループ外に生成するようにしたものである。

5

【００２０】例えば，最適化対象のプログラム１０において，ループ内にグローバルメモリ１７における配列データのアクセスＡ１，Ａ２，…があるとき，これらをシャドウを共通化するグループに分類し，転送範囲とシャドウの形状を決定する。そして，シャドウごとにグローバルメモリ１７から必要な配列データの一括受信を行う命令をループの前に付加し，また処理結果である配列データをグローバルメモリ１７へ一括送信する命令をループの後に付加する。ループ内ではローカルメモリ１８におけるシャドウをアクセスする。

【００２１】これにより，各ループ実行ごとにグローバルメモリ１７との間の要素単位のデータ転送が不要になり，一括転送によるデータ転送回数の削減および転送量の削減により，プログラムの高速実行が可能になる。ベクトル化できないようなケースでも，高速化を実現することができる。

【００２２】さらに，請求項２記載の発明では，データ依存解析をもとに受信が不要である配列要素を算出し，それを受信対象から外すので，一層の転送データ量の削減が可能になる。

【００２３】

【実施例】図２は，本発明の適用例説明図である。ソースファイル２１は，最適化対象となるソースプログラムが格納されたファイルである。構文解析部２２は，ソースファイル２１からソースプログラムを読み出し，所定の構文規則に従って構文を解析し，解析結果を中間コード２３として出力する。中間コード２３は，本計算機システムによって定められた内部形式データであり，実質的内容はソースプログラムと同等のものである。中間コード２３は，実際にファイルとして保持してもよく，またメモリ内に保持するようにしてもよい。

【００２４】ここで，本発明に係る最適化処理部１１による最適化を行う。前後に，他の最適化部２８ａ，２８ｂによる最適化を行ってもよい。ファイル出力部２４は，最適化処理後の中間コード２３をもとに，最適化されたソースファイル２５を出力する。これをコンパイラ２６の入力ファイルとしてコンパイルすることにより，データの一括転送を行うオブジェクトモジュール２７が生成される。

【００２５】図２に示す方式は，いわゆるプリコンパイラ方式による最適化の構成例であるが，プリコンパイラ方式によらずに，コンパイラ２６内部で最適化処理を実行するようにしてもよい。

【００２６】〔請求項１の発明の実施例〕図３は，請求項１記載の発明の実施例処理フローチャートである。すべてのグローバル変数G_i（$i=1, \cdots, m$）について，図３に示す(a)～(c)を実行し，最後に(d)を実行する。ここでグローバル変数とは，図１（Ｂ）に示すグローバルメモリ１７内に設けられる変数である。以下，この処理の流れに従って，本発明の実施例を説明する。

6

【００２７】(a) 先行順序関係の解析

同一の対象配列をアクセスする組（ｆ，ｇ）のうち，ｆ，ｇともに参照の場合を除くすべてについて，その先行順序関係を求める。先行順序関係は，次の４通りについて識別する。この先行順序関係の解析処理では，ベクトル化技術として知られている既存の技術を用いることができる。

【００２８】

①ｆ→ｇ　　アクセスｆがアクセスｇに常に先行する。

②ｇ→ｆ　　アクセスｇがアクセスｆに常に先行する。

③ｆ←→ｇ　アクセスｆがアクセスｇに先行するときと，逆のときがある。

【００２９】または，順序関係が不明である。

④ϕ　　アクセスされる配列要素に重なりがない。

(b) シャドウグループS_{ij}（$j=1, \cdots, n_i$）の決定

対象配列のアクセスを，シャドウを共通化するグループ（シャドウグループ）に分類する。図４は，そのシャドウグループの決定論理説明図である。先行順序が図４に示す○印の関係にある２つのアクセスは，必ず同一のグループに属するように分類する。例えば，アクセスｆがアクセスｇに常に先行し，ｆが定義，ｇが参照のアクセスである場合，この２つのアクセスは同一のグループとする。

【００３０】(c) シャドウ形状と転送範囲の決定

上記(b)で決定したすべてのシャドウグループS_{ij}について，以下の処理を行う。

【００３１】①　シャドウグループに属する参照および定義について，それぞれアクセス要素の集合和を求める。それがシャドウの受信および送信データの範囲となる。この受信データの範囲をΛ，送信データの範囲をΣで表す。

【００３２】②　ΛとΣの集合和を求める。これをシャドウ空間と呼ぶ。要素とシャドウ空間上の位置とのマッピングを示す関数かテーブルを求めておく。以下の説明では，このマッピングをＭで表す。

【００３３】シャドウ空間が重なり合っていて，共通化するとメモリ領域が節約できると判断できた場合，複数のシャドウを共通化して一つのシャドウとしてもよい。そのとき，ΛおよびΣは，それぞれもとのシャドウのΛおよびΣの集合和とする。

【００３４】(d) 出力コード生成

シャドウごとに，シャドウ空間とΛ，Σに応じたコード生成を行う。

①　シャドウ空間を動的または静的に確保するコードを生成する。必要なシャドウ空間を包含する大きめの領域を確保してもよい。

【００３５】②　グローバルメモリからシャドウへΛを受信するコードを，ループの直前に生成する。同様に，Σをシャドウからグローバルメモリへ送信するコードを，ループの直後に生成する。

7

【００３６】③ ループ内でのグローバルメモリへのアクセス部分は，マッピングMに応じて，シャドウのアクセスに変形する。次に，実際の最適化処理の例を説明する。図５は本発明の具体例説明図，図６は出力コードの例を示す図である。

【００３７】例えば図５（A）に示すようなプログラムを最適化するものとする。このプログラムは，ループ変数Ｊ，Ｉについての多重ループとなっている。配列Ａは，グローバル変数である。

【００３８】(a) 先行順序の解析

図５（A）に示す配列アクセスｕ１，ｕ４をｆとし配列アクセスｕ１～ｕ５をｇとして，前述した４通りのそれぞれの先行順序関係を調べると，図５（B）に示す先行順序関係の解析結果が得られる。例えば，アクセスｕ１とアクセスｕ２とは，それぞれＡ（Ｉ，Ｊ）の定義と，Ａ（Ｊ，Ｉ）の参照であり，ＩとＪの位置が逆転しているので，ｕ１が先行するときとｕ２が先行するときがある。また，アクセスｕ１とアクセスｕ３とは，ｕ３がＡ（Ｉ，Ｊ＋３０）であるので，ｕ１のＡ（Ｉ，Ｊ）より常に先行する。

【００３９】(b) シャドウグループの決定

図５（B）に示す先行順序関係の解析結果を，図４に示す共通化の決定論理にあてはめると，共通化が必須のアクセスのペアは，次の組となる。

【００４０】① ｕ１とｕ２

② ｕ１とｕ５

③ ｕ３とｕ４

よって，シャドウグループは，以下の２つとなる。

【００４１】① Ｓ１＝｛ｕ１，ｕ２，ｕ５｝

② Ｓ２＝｛ｕ３，ｕ４｝

(c) シャドウ形状と転送範囲の決定

シャドウグループＳ１について，次のようにシャドウ空間を決定する。

【００４２】

Λ＝（ｕ２のアクセス範囲）Ｕ（ｕ５のアクセス範囲）

＝｛(1,1),(2,1),…,(100,1), …,(100,100)｝

Σ＝（ｕ１のアクセス範囲）

＝｛(1,1),(2,1),…,(100,1), …,(100,100)｝

Λ∪Σ＝｛(1,1),(2,1),…,(100,1), …,(100,100)｝

この結果から，シャドウ領域として，ローカルメモリ上に二次元配列ａ１（１００，１００）を確保する。グローバルとシャドウは，同じ添字を持つ要素同士を対応づける。すなわち，Ａ（ｉ，ｊ）とａ１（ｉ，ｊ）とを対応づける。

【００４３】またシャドウグループＳ２について，次のようにシャドウ空間を決定する。

Λ＝（ｕ３のアクセス範囲）

＝｛(1,31),(2,31),…,(100,31),…,(100,130)｝

Σ＝（ｕ４のアクセス範囲）

＝｛(1,101),(2,101),…,(100,101), …,(100,200)｝

8

Λ∪Σ＝｛(1,31),(2,31),…,(100,31),…,(100,200)｝

この結果から，シャドウ領域として，二次元配列ａ２（１００，１７０）を確保する。グローバルとシャドウは，二次元目を３０ずらして対応づける。すなわち，Ａ（ｉ，ｊ）とａ２（ｉ，ｊ－３０）とを対応づける。

【００４４】(d) 出力コード生成

シャドウグループＳ１，Ｓ２について，シャドウ空間を確保するコードを生成し，グローバルメモリからシャドウへΛを受信するコードと，シャドウからグローバルメモリへΣを送信するコードを，それぞれループの前後に付加する。また，ループ内のアクセスを配列ａ１，ａ２へのアクセスに変換する。

【００４５】この結果，図６に示すような内容のコードが生成されることになる。すなわち，図５（A）に示すプログラムが，図６に示すような最適化されたプログラムに変換される。

【００４６】図６において，ａｌｌｏｃはローカルメモリの領域割り当て命令であり，「ａｌｌｏｃ ａ１（１００，１００）」は，１００×１００要素の二次元配列ａ１をローカルメモリに確保することを指示している。同様に，「ａｌｌｏｃ ａ２（１００，１７０）」は，１００×１７０要素の二次元配列ａ２をローカルメモリに確保することを指示している。

【００４７】ｒｅｃｅｉｖｅは，グローバルメモリからローカルメモリへのデータ転送（受信）命令であり，「ｒｅｃｅｉｖｅ ａ１（１：１００，１：１００）＝Ａ（１：１００，１：１００）」は，二次元配列のＡ（１，１）からＡ（１００，１００）までの全データを，ａ１（１，１）からａ１（１００，１００）までの領域に転送することを指示している。次の「ｒｅｃｅｉｖｅ ａ２（１：１００，１：１００）＝Ａ（１：１００，３１：１３０）」は，二次元配列のＡ（１，３１）からＡ（１００，１３０）までのデータを，ａ２（１，１）からａ２（１００，１００）までの領域に転送することを指示している。

【００４８】アクセスｕ１～ｕ５の命令は，図６に示すように配列ａ１，ａ２の対応する領域に対するアクセスに変更される。ｓｅｎｄは，ローカルメモリからグローバルメモリへのデータ転送（送信）命令であり，「ｓｅｎｄ Ａ（１：１００，１：１００）＝ａ１（１：１００，１：１００）」は，二次元配列のａ１（１，１）からａ１（１００，１００）までのデータを，Ａ（１，１）からＡ（１００，１００）までの領域に転送することを指示している。次の「ｓｅｎｄ Ａ（１：１００，１０１：２００）＝ａ２（１：１００，７１：１７０）」は，二次元配列のａ２（１，７１）からａ２（１００，１７０）までのデータを，Ａ（１，１０１）からＡ（１００，２００）までの領域に転送することを指示している。

【００４９】「ｆｒｅｅ ａ１」，「ｆｒｅｅ ａ２」

9

は，それぞれローカルメモリ上の配列ａ１，ａ２の領域を解放する命令である。

〔請求項２の発明の実施例〕図７は，請求項２記載の本発明の実施例処理フローチャートである。

【００５０】すべてのグローバル変数Ｇｉ（ｉ＝１，…，ｍ）について，図７に示す(a),(b),(e),(f)を実行し，最後に(d)を実行する。ここでグローバル変数とは，図１（Ｂ）に示すグローバルメモリ１７内に設けられる変数である。以下，この処理の流れに従って，本発明の実施例を説明するが，(a),(b),(d)の処理については請求項１記載の実施例と同様であるので，説明を省略する。

【００５１】(e) 受信転送範囲の縮小

処理(b)で決定したシャドウグループＳｉｊ（ｊ＝１，…，ｎｉ）の配列要素の集合のうち，受信転送範囲から除外することができるアクセス要素範囲を検出する。図８は，受信転送範囲の縮小の実施例の処理フローチャートである。

【００５２】図８において，シャドウグループＳｉｊに属する参照をｒｋ（ｋ＝１，…，Ｋ），シャドウグループＳｉｊに属する定義をｇｌ（ｌ＝１，…，Ｌ），参照または定義ｆでアクセスされる配列要素の集合をα(f)とし，すべての参照ｒについて，以下の処理を行う。

【００５３】① すべての参照ｒｋについて，参照ｒｋの受信転送範囲λ（ｒｋ）の初期値をα（ｒｋ）とする。

② 参照ｒｋと同一のシャドウグループに属し，先行順序がｇｌ→ｒｋの関係にあるかどうかを検出する。

【００５４】③ λ（ｒｋ）とα（ｇｌ）の集合差を改めてλ（ｒｋ）とする。すべての定義ｇｌについてこれを繰り返す。すなわち，ｋ＝１と初期化し（処理１００），処理１０８によってｋを１ずつ増加させて，ｋが参照の個数Ｋを超える（処理１０１）まで，以下の処理を行う。処理１０２により，ｋ番目の参照ｒｋの受信転送範囲λ（ｒｋ）の初期値をα（ｒｋ）とする。次に，同一のシャドウグループに属するｌ＝１からＬまでの定義ｇｌについて，先行順序が先行順序がｇｌ→ｒｋの関係にあるものを検出し，検出できたならば，λ（ｒｋ）とα（ｇｌ）の集合差を求めて，λ（ｒｋ）と置き換える（処理１０３～１０７）。

【００５５】なお，他の実施例として，参照ｒと同一のシャドウグループに属し，ｇ←→ｒの関係にある定義ｇを検出して，λ（ｒ）とα（ｇ）の共通要素のうち，最初のアクセスが定義であるものをさらにλ（ｒ）から削除するようにしてもよい。

【００５６】(f) シャドウ形状と転送範囲の決定

上記(e)の処理の後，すべてのシャドウグループＳｉｊについて，以下の処理を行う。

【００５７】① シャドウグループに属するすべての参照ｒの受信転送範囲λ（ｒ）の集合和をシャドウの受信転送データの範囲とする。これをΛと表すと，シャドウグループＳｉｊの受信転送範囲Λは，Λ＝λ（ｒ１）Ｕλ（ｒ２）Ｕ…Ｕλ（ｒｋ）となる。また，シャドウグループに属するすべての定義ｇのアクセス範囲α（ｇ）の集合和をシャドウの送信転送範囲とする。以下，これをΣと表す。

10

【００５８】② ΛとΣの集合和を求める。これをシャドウ空間と呼ぶ。要素とシャドウ空間上の位置とのマッピングを示す関数かテーブルを求めておく。シャドウ空間が重なり合っていて，共通化するとメモリ領域が節約できると判断できた場合，複数のシャドウを共通化して一つのシャドウとしてもよい。そのとき，ΛおよびΣは，それぞれもとのシャドウのΛおよびΣの集合和とする。

【００５９】次に，実際の最適化処理の例を前述の図５を用いて説明する。例えば図５（Ａ）に示すようなプログラムを最適化するものとする。このプログラムは，ループ変数Ｊ，Ｉについての多重ループとなっている。配列Ａはグローバル変数である。

【００６０】（ａ）先行順序の解析

図５（Ａ）に示す配列アクセスｕ１，ｕ４をｆとし配列アクセスｕ１～ｕ５をｇとして，前述した４通りのそれぞれの先行順序関係を調べると，図５（Ｂ）に示す先行順序関係の解析結果が得られる。例えば，アクセスｕ１とアクセスｕ２とは，それぞれＡ（Ｉ，Ｊ）の定義と，Ａ（Ｊ，Ｉ）の参照であり，ＩとＪの位置が逆転しているので，ｕ１が先行するときとｕ２が先行するときがある。また，アクセスｕ１とアクセスｕ３とは，ｕ３がＡ（Ｉ，Ｊ＋３０）であるので，ｕ１のＡ（Ｉ，Ｊ）より常に先行する。

【００６１】(b) シャドウグループの決定

図５（Ｂ）に示す先行順序関係の解析結果を，前述の図４に示す共通化の決定論理にあてはめると，共通化が必須のアクセスのペアは，次の組となる。

【００６２】① ｕ１とｕ２

② ｕ１とｕ５

③ ｕ３とｕ４

よって，シャドウグループは，以下の２つとなる。

【００６３】① Ｓ１＝{ｕ１，ｕ２，ｕ５}

② Ｓ２＝{ｕ３，ｕ４}

(e) 受信転送範囲の縮小

シャドウグループＳ１についての受信転送範囲は，次のようになる。

【００６４】参照ｕ２については，定義ｕ１との先行順序関係がｕ１←→ｕ２である。参照ｕ２に対し，ｕ１の要素の一部が先行するが，パターンが複雑と判断し無視する。受信範囲の縮小化はここでは行わない。

【００６５】λ（ｕ２）＝α（ｕ２）

＝{(1,1),(2,1),…,(100,1),…,(100,100)}

参照ｕ５に対して，定義ｕ１が常に先行する。

11

【0066】λ（u5）＝α（u5）－α（u1）＝φ

一方，シャドウグループS2については，次のようになる。参照u3に対して，定義u4が常に先行する。

【0067】λ（u3）＝α（u3）－α（u4）

＝{(1,31),(2,31),…,(100,31),…,(100,100)}

(f) シャドウ形状と転送範囲の決定

シャドウグループS1について，次のようにシャドウ空間を決定する。

【0068】Λ＝λ（u2）∪λ（u5）

＝{(1,1),(2,1),…,(100,1), …,(100,100)}

Σ＝α（u1）

＝{(1,1),(2,1),…,(100,1), …,(100,100)}

Λ∪Σ＝{(1,1),(2,1),…,(100,1), …,(100,100)}

この結果から，シャドウ領域として，ローカルメモリ上に二次元配列a1（100，100）を確保する。グローバルとシャドウは，同じ添字を持つ要素同士を対応づける。すなわち，A（i，j）とa1（i，j）とを対応づける。

【0069】またシャドウグループS2について，次のようにシャドウ空間を決定する。

Λ＝λ（u3）

＝{(1,31),(2,31),…,(100,31),…,(100,100)}

Σ＝α（u4）

＝{(1,101),(2,101),…,(100,101), …,(100,200)}

Λ∪Σ＝{(1,31),(2,31),…,(100,31),…,(100,200)}

この結果から，シャドウ領域として，二次元配列a2（100，170）を確保する。グローバルとシャドウは，二次元目を30ずらして対応づける。すなわち，A（i，j）とa2（i，j－30）とを対応づける。

【0070】(d) 出力コード生成

シャドウグループS1，S2について，シャドウ空間を確保するコードを生成し，グローバルメモリからシャドウへΛを受信するコードと，シャドウからグローバルメモリへΣを送信するコードを，それぞれループの前後に付加する。また，ループ内のアクセスを配列a1，a2へのアクセスに変換する。

【0071】この結果，図9に示すような内容のコードが生成されることになる。すなわち，図5（A）に示すプログラムから，図9に示すような最適化されたプログラムが得られる。

【0072】前述の実施例における図6の出力コードとの違いは，シャドウグループS2についての受信転送範囲Λが，図6では二次元配列のA（1，31）からA（100，130）までの領域であるのに対し，本実施例では，A（1，31）からA（100，100）までの領域であり，受信転送範囲が縮小されていることである。

【0073】なお，上記(e) の受信転送範囲の縮小処理において，シャドウグループS1とS2を共通化する方がよいと判断できた場合には，シャドウ空間を次のようにしもよい。

12

【0074】Λ＝Λ（S1）∪Λ（S2）

＝{(1,1),(2,1),…,(100,1), …,(100,100)}

Σ＝Σ（S1）∪Σ（S2）

＝{(1,1),(2,1),…,(100,1), …,(100,200)}

Σ∪Λ＝{(1,1),(2,1),…,(100,1), …,(100,200)}

この結果から，シャドウ領域として，ローカルメモリ上に二次元配列a（100，200）を確保する。グローバルとシャドウは，同じ添字を持つ要素同士を対応づける。すなわち，A（i，j）とa（i，j）とを対応づける。

【0075】この結果，図10に示すような内容のコードが生成されることになる。すなわち，受信転送範囲Λは，Λ（S1）∪Λ（S2）により，二次元配列のA（1，1）からA（100，100）までの領域が対象となる。また，送信転送範囲Σは，Σ（S1）∪Σ（S2）により，A（1，1）からA（100，200）までの領域が対象となる。

【0076】図11は，本発明の適用結果の例を示す図である。例えば図11（A）に示すようなプログラムについて，請求項1記載の発明によってデータ転送の一括化を図った場合，出力コードは図11（B）に示すようになる。また，請求項2記載の発明によって，さらに受信転送範囲の縮小化を行った場合，その結果は図11（C）に示すようになる。

【0077】図11（A）に示す元のプログラムから明らかなように，A（k）（ただし，k＝2，3，4，…，）の定義は，A（k）の参照より先行する。したがって，A（2），A（3），…の受信は不要であり，図11（C）の例では，グローバルメモリから受信するデータがA（1）だけになっている。

【0078】図12は，本発明が適用可能なシステム構成の例を示す図である。本発明によるデータ転送の一括化処理方法は，例えば図12（A）～（C）に示すような2階層のメモリ構造を持つシステムで動作するプログラムを対象として適用することができる。

【0079】図12（A）のシステムは，共有メモリ4を各プロセッサ・エレメント1が共有する並列計算機システムである。各プロセッサ・エレメント1は，CPU2とローカルメモリ3からなる。CPU2が実行するプログラムのループ中に，共有メモリ4上の配列に対するアクセスがあるとき，本発明の適用により，その配列データをループの直前で共有メモリ4からローカルメモリ3へ受信するようにし，ループの直後でローカルメモリ3から共有メモリ4へ送信するようにすることにより，プログラムの高速化が可能になる。

【0080】図12（B）のシステムは，仮想的共有メモリを持つ並列計算機または分散計算機環境のシステムであって，各プロセッサ・エレメント1は，ネットワーク6で接続されている。各プロセッサ・エレメント1

13

は，ＣＰＵ２とローカルメモリ３と他のプロセッサがアクセスできるメモリ５を持つ。各プロセッサ・エレメント１に分散して配置されたメモリ５は，例えば全体で一つのメモリ空間を構成するように扱われる。ＣＰＵ２は，メモリ５のデータにアクセスする必要があるとき，メモリ５のデータをローカルメモリ３に複写して，ローカルメモリ３上でアクセスする。このような他のプロセッサがアクセスできるメモリ５を，図１に示すグローバルメモリ１７として，本発明を適用することができる。

【００８１】図１２（Ｃ）のシステムは，大容量外部メモリ装置７を持つ計算機システムである。プロセッサ・エレメント１は，ＣＰＵ２とローカルメモリ３とを持ち，大容量外部メモリ装置７上のデータを参照・更新する場合には，ローカルメモリ３にデータを複写してアクセスする。このようなローカルメモリ３と大容量外部メモリ装置７からなる２階層メモリのシステムにも，同様に本発明を適用し，プログラム実行の高速化を図ることができる。

【００８２】

【発明の効果】以上説明したように，本発明によれば，アクセスの一括化により転送回数および転送総量の削減が可能になり，データ転送に関するオーバヘッドが削減される。また，対象変数の先行順序関係のみで転送対象を共通化して抽出することができるので，従来のベクトル化技術におけるベクトル可否等の判断よりも容易であり，効果も大きい。また，転送を多重ループの外に出すことができるため，さらに転送のスケジューリングなどの最適化が可能になる。例えば，非同期転送が可能なハードウェアであれば，受信転送をなるべく早く開始し，送信転送の終了待ちをなるべく遅らせることにより，計算と転送のオーバラップによる処理の高速化を図ることができるようになる。

14

【図面の簡単な説明】

【図１】本発明の原理説明図である。

【図２】本発明の適用例説明図である。

【図３】本発明の実施例処理フローチャートである。

【図４】シャドウグループの決定論理説明図である。

【図５】本発明の具体例説明図である。

【図６】本発明の実施例による出力コードの例を示す図である。

【図７】本発明の実施例処理フローチャートである。

【図８】受信転送範囲縮小の実施例処理フローチャートである。

【図９】本発明の実施例による出力コードの例を示す図である。

【図１０】本発明の実施例による出力コードの例を示す図である。

【図１１】本発明の適用結果の例を示す図である。

【図１２】本発明が適用可能なシステム構成の例を示す図である。

【図１３】２階層メモリのモデル説明図である。

【図１４】本発明の課題説明図である。

【符号の説明】

１０　プログラム
１１　最適化処理部
１２　アクセス先行順序関係解析処理部
１３　シャドウグループ決定処理部
１４　転送範囲決定処理部
１５　コード生成処理部
１６　最適化されたプログラム
１７　グローバルメモリ
１８　ローカルメモリ
１９　プロセッサ

【図４】

シャドウグループの決定論理説明図

	f：参照 g：定義	f：定義 g：参照	f：定義 g：定義
f→g	－	○	○
f←→g	○	○	○
g→f	○	－	○
φ	－	－	－

【図５】

具体例説明図

(A)

```
DO J=1,100
   DO I=1,100
      A(I,J)= …          u1
      … =A(J,I)          u2
      … =A(I,J+30)       u3
      A(I,J+100)= …      u4
      … =A(I,1)          u5
   END DO
END DO
```

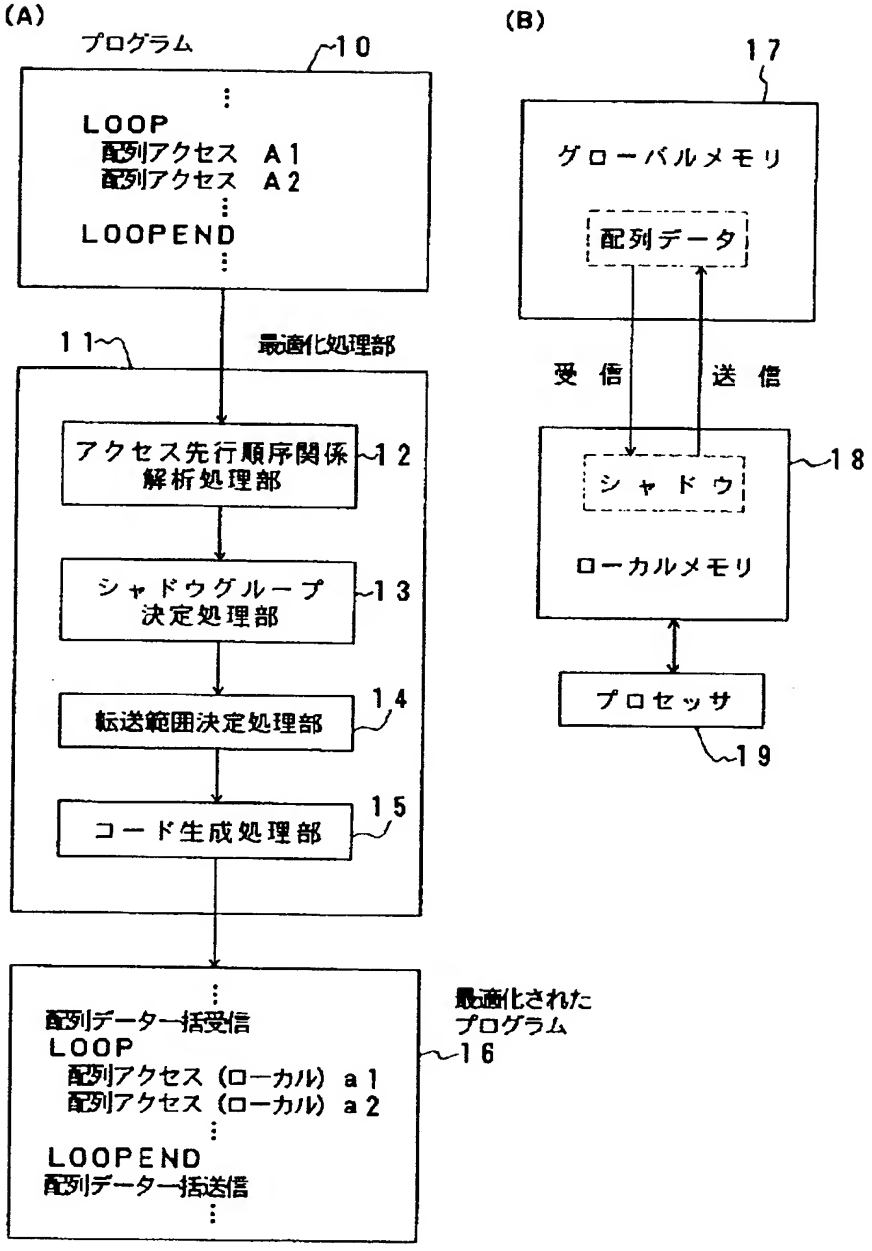
(B)

先行順序関係の解析

f ＼ g	u1	u2	u3	u4	u5
u1	φ	f←→g	g→f	φ	f→g
u4	φ	φ	f→g	φ	φ

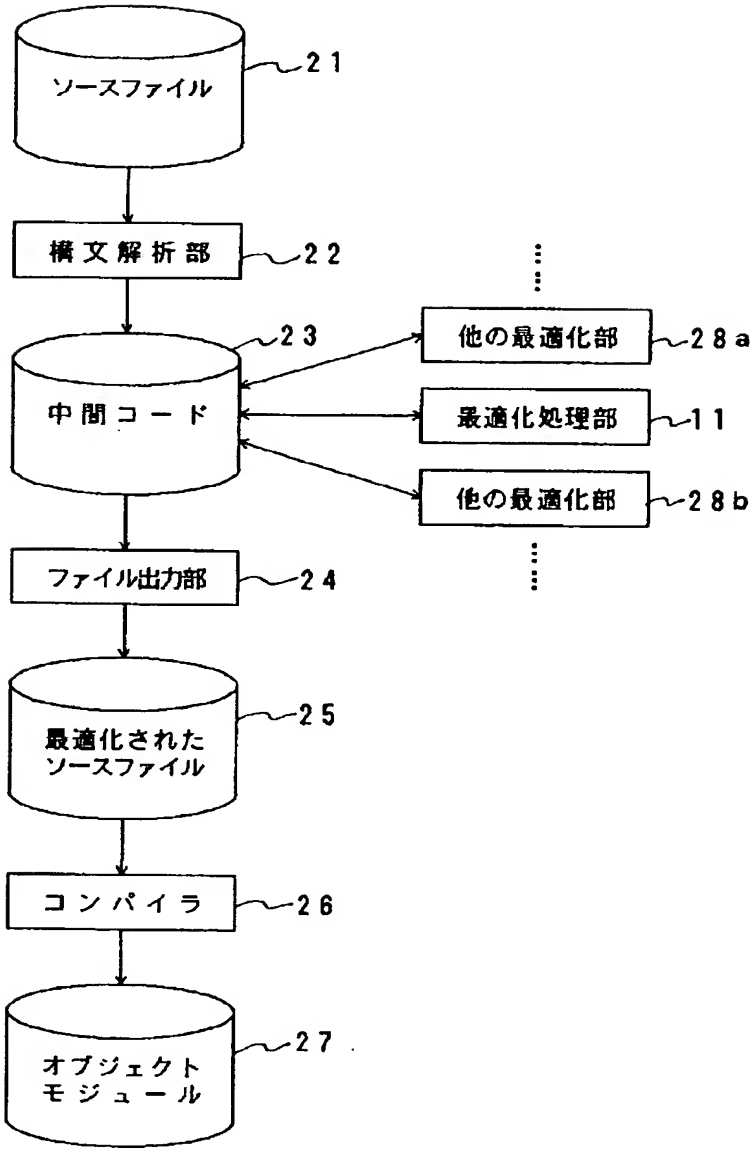
【図1】

本発明の原理説明図



【図２】

本発明の適用例説明図



【図１１】

本発明の適用結果の例

(A)

```
DO i=1,100
  c(i)=A(i)+b(i)
  A(i+1)=c(i)
END DO
```

(B)

```
receive tmp(1:100)=A(1:100)
DO i=1,100
c(i)=tmp(i)+b(i)
tmp(i+1)=c(i)
END DO
send A(2:101)=tmp(2:101)
```

(C)

```
receive tmp(1:1)=A(1:1)
DO i=1,100
c(i)=tmp(i)+b(i)
tmp(i+1)=c(i)
END DO
send A(2:101)=tmp(2:101)
```

【図１４】

本発明の説明図

(a)

```
DO i=1,100
  c(i)=A(i)+b(i)
  A(i-1)=c(i)
END DO
```

(b)

```
c(1:100)=A(1:100)+b(1:100)
A(0:99)=c(1:100)
```

(c)

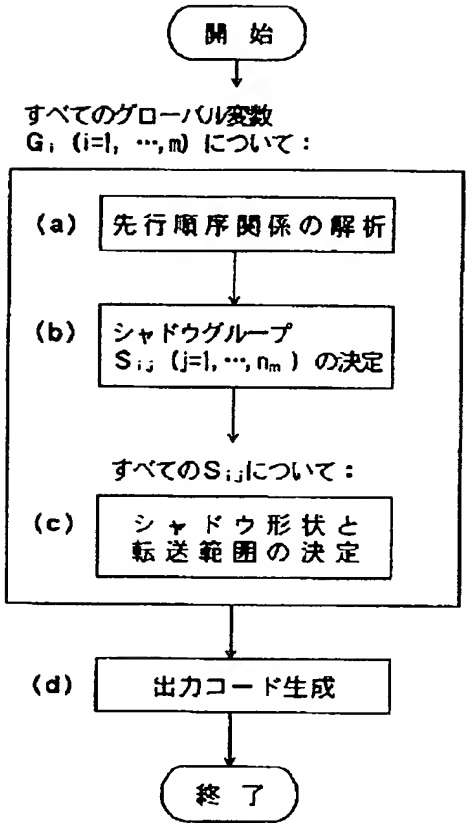
```
receive tmp1(1:100)=A(1:100)
c(1:100)=tmp1(1:100)+b(1:100)
tmp2(0:99)=c(1:100)
send A(0:99)=tmp2(0:99)
```

(d)

```
DO i=1,100
  c(i)=A(i)+b(i)
  A(i+1)=c(i)
END DO
```

【図3】

実施例処理フローチャート



【図6】

出力コードの例

```
alloc a1(100,100)
alloc a2(100,170)

receive a1(1:100,1:100)=A(1:100,1:100)        Λ (S1)
receive a2(1:100,1:100)=A(1:100,31:130)       Λ (S2)

DO J=1,100
  DO I=1,100
    a1(I,J)=…                                 u1
    … = a1(J,I)                               u2
    … = a2(I,J)                               u3
    a2(I,J+70)= …                             u4
    … = a1(I,1)                               u5
  END DO
END DO

send A(1:100,1:100)=a1(1:100,1:100)           Σ (S1)
send A(1:100,101:200)=a2(1:100,71:170)        Σ (S2)

free a1
free a2
```

【図10】

出力コードの例

```
alloc a(100,200)

receive a(1:100,1:100)=A(1:100,1:100)         Λ (S1) UΛ (S2)

DO J=1,100
  DO I=1,100
    a(I,J)= …                                 u1
    … = a(J,I)                                u2
    … = a(I,J+30)                             u3
    a(I,J+100)= …                             u4
    … = a(I,1)                                u5
  END DO
END DO

send A(1:100,1:200)=a(1:100,1:200)            Σ (S1) UΣ (S2)

free a
```

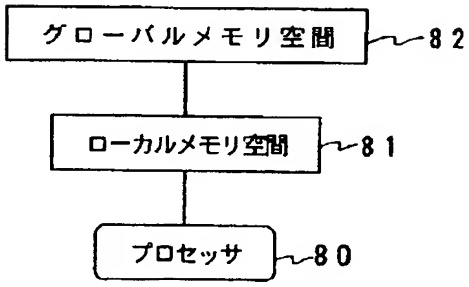
【図9】

出力コードの例

```
alloc a1(100,100)
alloc a2(100,170)

receive a1(1:100,1:100)=A(1:100,1:100)        Λ (S1)
receive a2(1:100,1:70)=A(1:100,31:100)        Λ (S2)

DO J=1,100
  DO I=1,100
    a1(I,J)=…                                 u1
    … = a1(J,I)                               u2
    … = a2(I,J)                               u3
    a2(I,J+70)= …                             u4
    … = a1(I,1)                               u5
  END DO
END DO

send A(1:100,1:100)=a1(1:100,1:100)           Σ (S1)
send A(1:100,101:200)=a2(1:100,71:170)        Σ (S2)

free a1
free a2
```

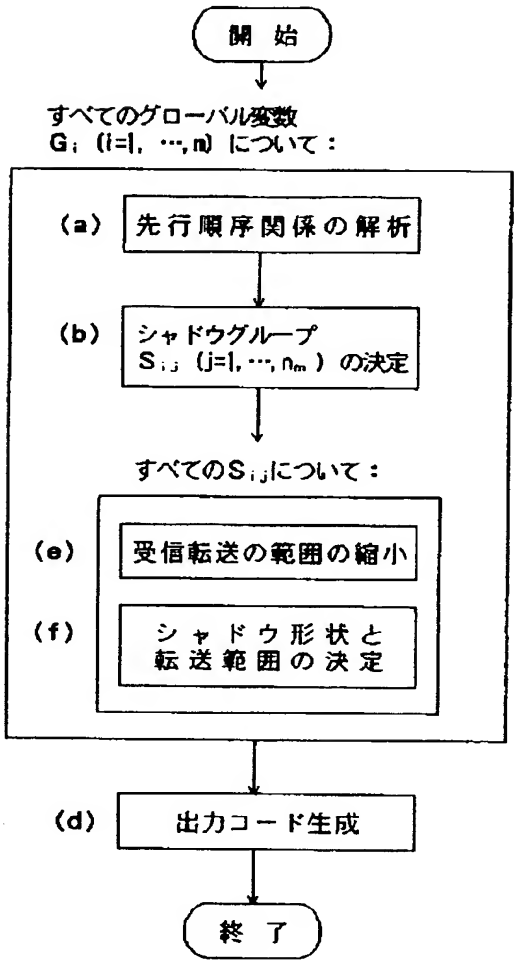
【図13】

2階層メモリのモデル説明図



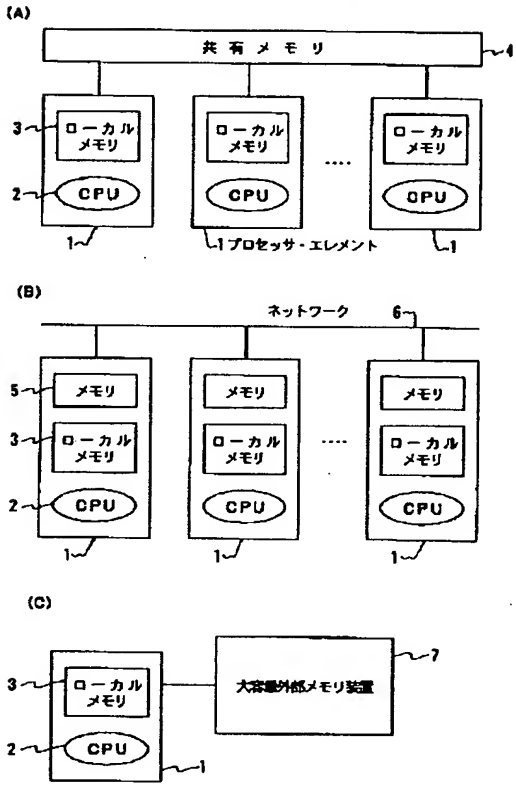
【図７】

実施例処理フローチャート



【図１２】

本発明が適用可能なシステム構成の例



【図8】

受信転送範囲縮小のフローチャート